Bullseye Case Study

PROG1197i Individual Software Development Project (ISDP)

Specifications Manual

Winter 2023

Chris London & Aaron Mitchell

Last Update: Dec 31, 2022

Contents

Introduction

Welcome to the show! 😊

This is your introduction to the big tamale, the big kahuna, the end game – the capstone project! This capstone project is based on a case study in which you must build a fully functional software system from a combination of given set of specifications and your own creativity and knowledge/skills.

This course is an opportunity for students to gain practical experience in system development. Students apply systems design theory and computer programming skills to complete a small systems development project under the supervision of an instructor.

Student progress is closely monitored by the instructor(s) using intermittent technical reviews and manual submissions. Students program, test, document and deliver a realistic, small-scale system with components written in a variety of languages against an approved Relational Database Management System (RDMS). Emphasis is placed upon ***individual initiative, resourcefulness, and self-discipline*** to build and implement this project from start to finish.

This document contains the guidelines for your project. It is not all-inclusive, there is plenty of room for you to add your own functionality and creativity, but the overall expectation is that your final project will perform the required features as designed with the goal of providing the end users a properly working system.

To ensure we can assess - and you can demonstrate - the range of your abilities, ***you will create a combination of a desktop and web-based app***, using the **provided MySQL database (bullseyedb2023_1.0).** If this database script needs to be tweaked along the way, the version will be updated at the end and you will be notified (bullseyedb2023_1.1, bullseyedb2023_1.2, etc.).

You may **add new tables, triggers, stored procedures, etc.,** to the database to handle any functionality you may wish to add, but **you may not alter existing tables *or data therein***.

If you add new tables/triggers/etc., you must create them using a separate .sql file to run ***after running the file provided***, as we will be running the provided file prior to all sprint reviews to ensure we have a valid database and my test data runs as expected

Expectations

- Discipline, product ownership, professionalism, dedication to the task at hand
- Efficient coding practices
- Proper commenting
- All code and documentation will be stored in a private GitHub library, which will be shared ONLY with your instructors (via email invite)
- Weekly log in which you journal your progress
- Four (4) sprints with a series of features/requirements to be completed, as listed in this document
- **Assessment:**
    - Log/Journal 5%
    - Sprint 1 20%
    - Sprint 2 25%
    - Sprint 3 25%
    - Sprint 4 25%
- Formal Sprint Planning sessions will occur at the start of each sprint (attendance is mandatory)
- Weekly demonstrations of your progress will occur during scheduled class times
- Formal Sprint Reviews will occur at the end of each sprint (attendance is mandatory)
- Grades based on functionality based on given specifications. If you are unsure of what is meant in a specification, ASK. Getting the specification wrong because you did not ask is not an excuse
- No grades are given for inefficient code, commenting, poor UX / design. At this point in your career, these are expected and while you will not receive credit for doing it, you will be penalized for not following specs, not writing efficient code, not commenting, and not following good UX / design standards
- Every sprint is REQUIRED to successfully complete this course (i.e. you cannot decide not to submit a sprint and be successful)
- This is an **INDIVIDUAL** project. There will be **NO SHARING CODE**. You may only share "***concepts***" with your classmates in an effort to help them out of a jam.
    - (Example: Someone having trouble connecting to a DB with a data table, you could help by giving them advice like "I solved that by using this library _____" or "Did you remember to do this ___".

Security and Permissions

The Bullseye Inventory Management System (BIMS) has permission-based access. To access a particular aspect of the system, the user logged in must have the appropriate permission(s). Each feature/functionality will have the appropriate permission(s) listed with the other specifications. Most of these permissions are self-explanatory.

System Permissions:

ACCEPTSTOREORDER
ADDITEMTOBACKORDER
ADDNEWPRODUCT
ADDSITE
ADDUSER
CREATEBACKORDER
CREATELOSS
CREATEREPORT
CREATESTOREORDER
CREATESUPPLIERORDER
DELETELOCATION
DELETEUSER
DELIVERY
EDITINVENTORY
EDITITEM
EDITPRODUCT
EDITSITE
EDITUSER
FULFILSTOREORDER
MODIFYRECORD
MOVEINVENTORY
PREPARESTOREORDER
PROCESSRETURN
READUSER
RECEIVESTOREORDER
SETPERMISSION
VIEWORDERS

Sprint 1

Sprint 1 is designed to set up the basics and lay the groundwork for the entire system. The following requirements are included in Sprint 1:

GitHub Project Repository

1. Create a new ***private*** repository in your GitHub account (if you do not have a GitHub account or cannot remember the one you created in Dev Tools, create a new one and use that).

2. Name your new repository as follows:
    a. **isdp2023_yourfirstnameyourlastname** (Example: **isdp2023_ChrisLondon**)

3. Invite your instructors to have access to your repository. You can do this inside your GitHub repository by going to Settings > Collaborators > Add by email

    Invite your instructors by using the following addresses:

    a. Chris.London@nbcc.ca
    b. Aaron.Mitchell@nbcc.ca

4. This repository will house ***ALL files associated with this project***, including this document and any document(s) you create, .sql files, source code, web pages, config files, notes, etc. This is your backup. “My hard drive crashed” or “My laptop died” are not excuses for losing your work. Keep your repository updated regularly. This is a work/life lesson that will serve you well in this industry.

Data Access Diagram

Create a Data Access Diagram (ER Diagram) using the existing Database Schema. You can obtain this via the MySQL Workbench. Example of how to do this can be found here:

https://medium.com/@tushar0618/how-to-create-er-diagram-of-a-database-in-mysql-workbench-209fbf63fd03

Remember to upload this to your GitHub project repository.

Data Dictionary

A data dictionary is a documented collection of data about the database and each table and field within, including names, definitions, default values, attributes, acceptable values. This is important so that you document each table so you understand the various fields, data types, sizes, etc.

Example data dictionary for a table called "employee":

Table: Employee						
Field	**Type**	**Format**	**Size**	**Description**	**Required?**	**Key?**
empID	INT	999999	6+	Auto-generated unique identifier for each customer. First value starts at 100001	Y	PK
firstName	varchar(32)	Text	50	Customer First Name	Y	
lastName	varchar(32)	Text	50	Customer Last Name	Y	
startDate	date	YYYY-MM-DD		Date customer account was created. Default: Today	Y	
street	varchar(32)		50		N	
city	varchar(32)		50		N	
province	varchar(32)		2	Link to province table. Default: NB	Y	FK
postcode	varchar(32)	L9L9L9	6	Only store alphanumeric values. Display as L9L 9L9	N	
phone	varchar(32)	5065551212	9	Only store numeric values. Display as (506) 555-1212	N	
email	varchar(32)	x@x.ca	50		N	
…						

Security

All employees using the system must have an active account and password (see employee table)

Password must be hidden on login form(s) and encrypted in DB (i.e. not stored in plain text) with AES 256 bit encryption

When viewed in the db, the password will be encrypted and NOT in plain text

Audit Activity

Every action in the system creates a record of that activity in the Audit table, so you need to develop this aspect of the system now so you don't have to backtrack later.

Info to be tracked in each audit record includes:

- txnAuditID – unique, autonumbered ID
- txnID – id of the transaction being audited (if applicable)
- txnType – type of transaction (if applicable)
- siteID – site where activity originated
- employeeID – employee who initiated the activity
- status – status of the transaction (if applicable)
- date/time – current datetime when this record is created
- description of the activity – a clear text description of what is happening
    - Examples:
        - Add new employee record jsmith
        - Edit employee record jsmith

A trigger or an audit object that takes in the parameters above and writes them to the txnAudit table in the DB. However you do it, remember: ***it needs to be available for any transaction or activity that occurs in the system***. Think of this as the logs, breadcrumbs required to trace activity, and a way to any potential issues or improper activity.

User Permissions

Admin requires the ability to set and remove user permissions in the system.

Methods:

- Position-based assignment
    - Each new user created gets assigned the appropriate permissions required for the position they are assigned (see ***Add User***, ***Edit User***)
- Individual permissions
    - Each user can have additional permissions (beyond those assigned to their role)

Actor(s): ADMIN

Permission(s): SETPERMISSION

Sample: None

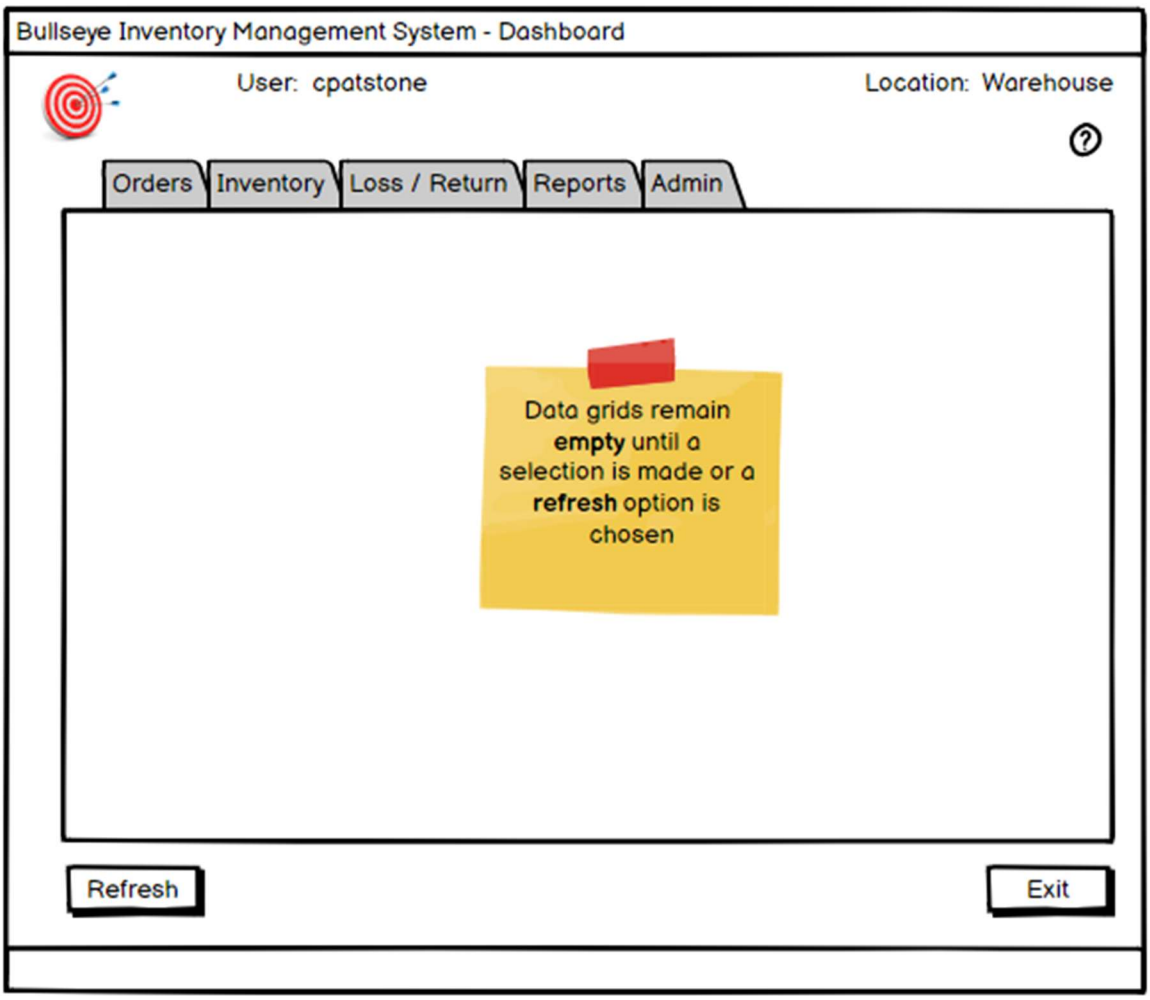
Dashboard

A dashboard that provides access to all aspects of the system, based on who is logged in.

Actor(s): ALL

Permission(s): ALL* (access to various areas/functionality is determined by permissions)

Sample:



Login

Every Bullseye user must have:

- A valid Username and password (first initial, last name - i.e. jperez, as described in the DB)
- Ability to recover username and/or password via email (we will mimic this with a form like the one below)
- permissions to perform specific actions
- Password encrypted in DB (i.e. not in plain text) – AES 256 bit
- Password rules: minimum 8 characters, start with a letter, contains at least 1 capital letter and 1 special character

A user who logs in with the correct username and password will be presented with the dashboard and have access to the appropriate functionality.

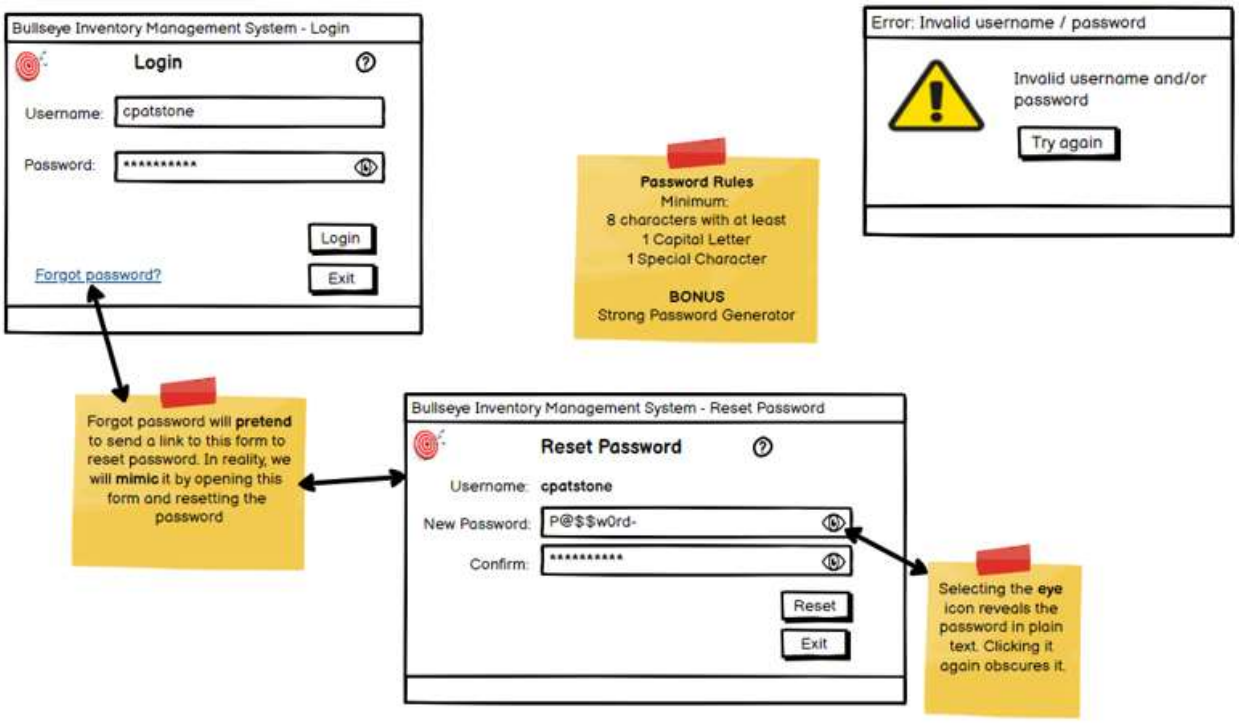
Any attempt to log in with an account that is inactive should fail (with appropriate warning)

Any attempt to log in with a username/password combination that is not accurate should fail (with appropriate warning)

Actor(s): ALL

Permission(s): None * (all active users accounts can log in)

Sample:



Logout

Every active Bullseye user can logout if they are currently logged in.

Logout methods:

1. Manual (select 'Exit' button from dashboard)
2. Automatic (after a period of inactivity, a time value which can be modified, default is 15 minutes). You can use either a config file or a database table to set this value.

Actor(s): ALL

Permission(s): None * (all active users accounts can log out)

Sample: None

Add User

Admin requires the ability to add a user to the system.

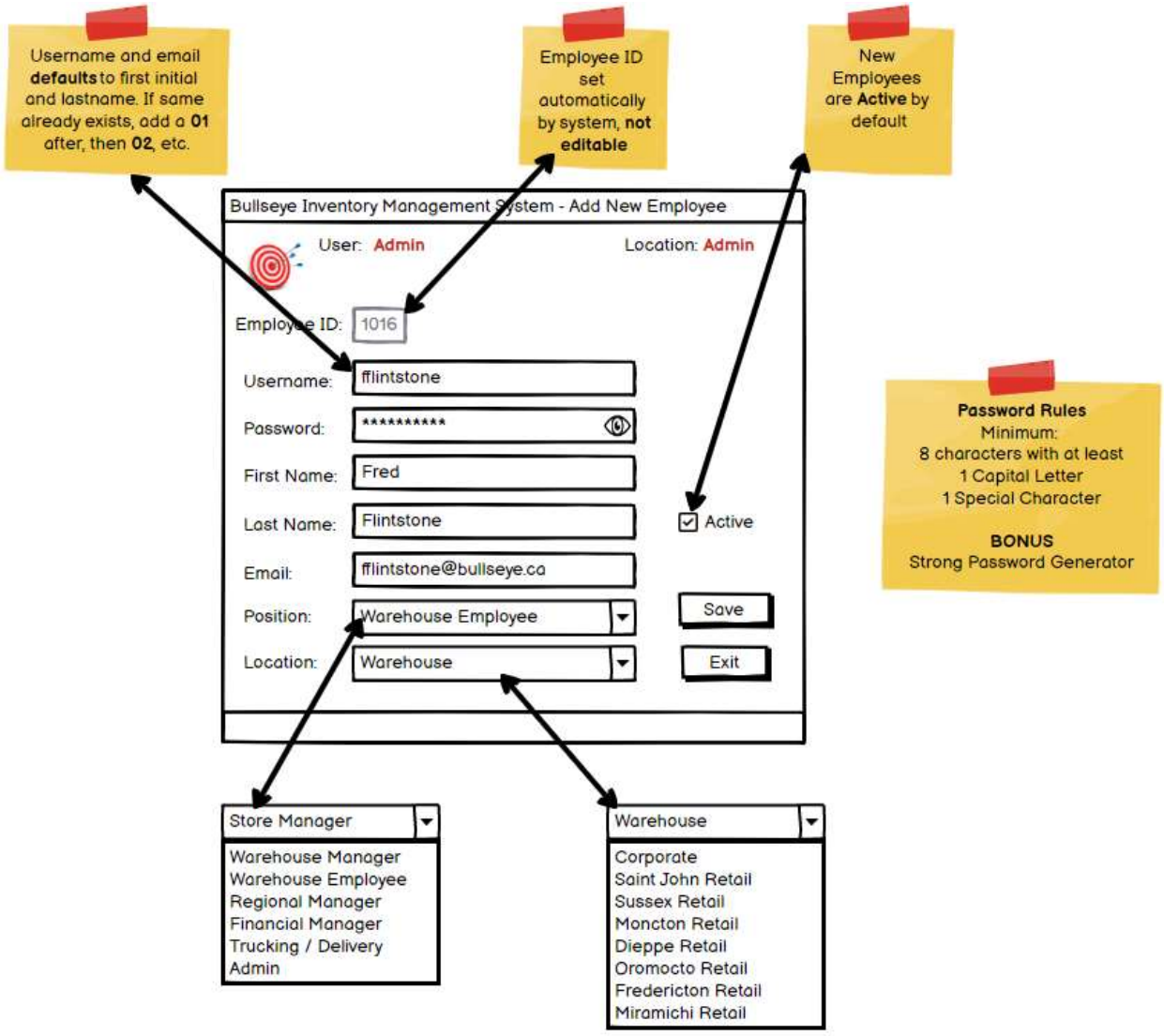
This will normally be triggered by a new user joining the company.

New user accounts are set “active” by default (the system sets their “active” status to TRUE or 1 in employee table).

Actor(s): ADMIN

Permission(s): ADDUSER

Sample:



Edit User

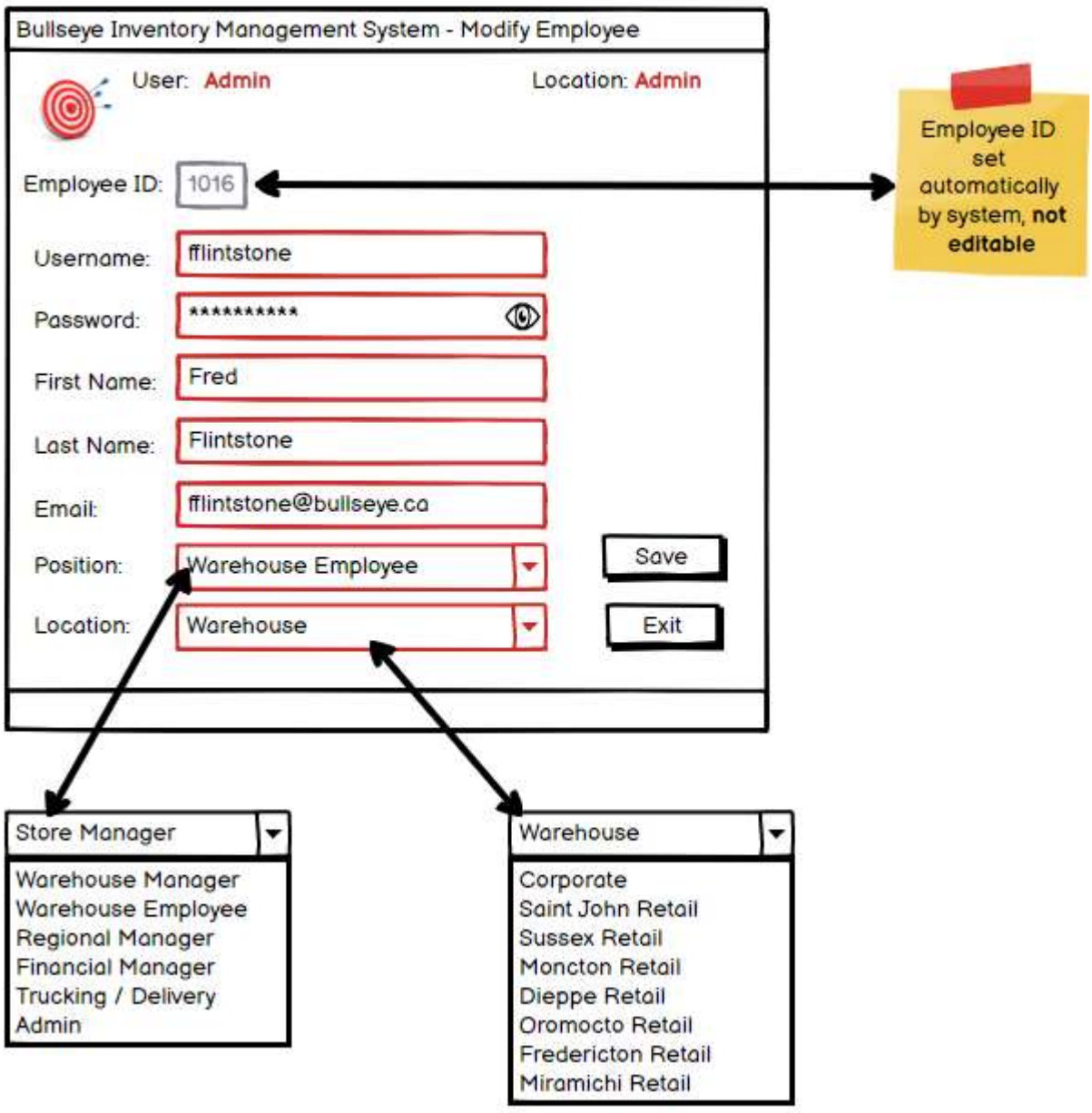
Admin requires the ability to edit a user's data in the system.

This will normally be triggered by a user changing jobs, getting married, etc.

Actor(s): ADMIN

Permission(s): EDITUSER

Sample:



Delete User

Admin requires the ability to remove a user from the system.

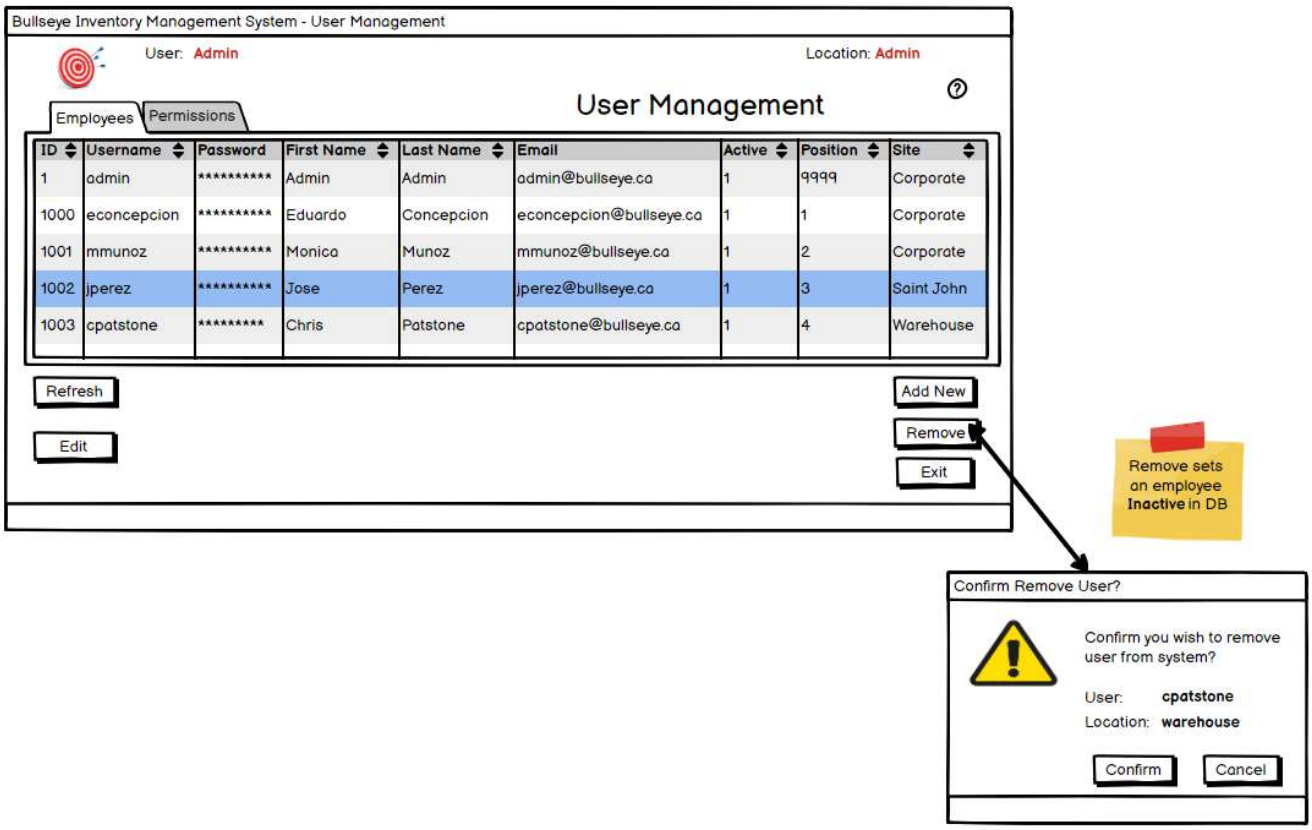
This will normally be triggered by a user leaving the company.

Any user “removed” is NOT deleted, the system simply sets their “active” status to FALSE (0 in employee table).

Actor(s): ADMIN

Permission(s): DELETEUSER

Sample:



Sprint 2

Create Store Order

Create Emergency Order

Receive Store Order

Prepare Store Order

Fulfil Store Order

Add Item to Backorder

View Store Order

Add Location

Edit Inventory

Move Inventory

Sprint 3

Pickup Store Order

Transport Store Order

Deliver Store Order

Place Online Order

Prepare Online Order

Receive Online Order

Accept Store Order

Modify Record *

Sprint 4

Create Loss

Process Return

Add New Product

Edit Product